Experiment 2: Wireshark TCP Analysis

1 Introduction

In this lab you will use Wireshark to investigate the behavior of the *transmission control protocol* (TCP), the primary transport layer protocol used to provide reliable data tranfser.

At the end of this lab you should know how to:

1. Use Wireshark features to analyze TCP traces;
2. Identify different phases and events of TCP congestion control in packet traces (slow start, congestion avoidance, various packet loss events);
3. Use data gathered in packet traces to quantify the steady-state throughput of a TCP connection.

As always, read the instructions carefully!

2 Background

Wireshark can be used to capture packet traces, apply filters to focus in on a desired subset of packets in a trace, and inspect the contents of individual packets in a trace. In this lab we will explore additional features of Wireshark that are especially useful for understanding and analyzing the performance and behavior of TCP. The Wireshark webpage www.wireshark.org provides a user guide and other very useful resources.

This lab assumes that you have an understanding of the various aspects of TCP discussed in ECSE 416 and covered in Chapter 3 of Kurose and Ross. This includes TCP's congestion control mechanism, three-way handshake and connection teardown mechanisms, factors affecting the theoretical steady-state throughput achieved by TCP, and TCP fairness.

3 Lab Requirements

3.1 TCP Wireshark Primer

DO THIS SECTION IF (AND ONLY IF) YOU ARE UNFAMILIAR WITH USING WIRESHARK TO ANALYZE TCP TRACES. Complete the "TCP" Wireshark lab in the supplemental document posted on myCourses. Example Wireshark traces are provided on myCourses so you do not need to download the *wireshark_traces.zip* as explained in the "TCP" lab.

3.2 TCP in the Wild

Collect at least four TCP traces, using Wireshark, of relatively long duration (*recommended:* at least two minutes each) from different servers with different latency and loss rates. For example, consider downloading a Linux distribution from different servers located around the world. Try to select servers from a variety of locations; servers in Africa, Asia, or South America may exhibit larger RTTs and more dropped packets.

Read the articles by Mathis et al. [1] and Padhye et al. [2] for background on different approaches to analyzing the steady-state throughput of TCP.

Next, using these traces, complete the following tasks:

1. Compare the observed TCP congestion window behavior to the theoretical behavior discussed in ECSE 416. Use Wireshark analysis tools to generate graphs or import the data into Excel or Matlab for graphing.
   a. Make sure you can identify a slow-start phase.
   b. Identify the congestion avoidance stage.
   c. Identify losses in the traces and classify the type of loss. *Hint:* Look at sequence numbers and check for re-transmissions and out-of-order packets.
   d. Determine whether there is a bound on the maximum congestion window size specified by the client or server.
   e. Assess what factors are limiting the throughput of the connection.

   You will be asked to show your results during the demo. You may prepare material ahead of time (screenshots, printed figures & tables) or you may reproduce your actions during the demo.

2. Estimate the empirical steady-state throughput of the long-duration TCP connections. Compare these estimates to values derived from the simple model developed by Mathis et al. [1],

   $$R = \frac{MSS}{RTT}\sqrt{\frac{3}{2p}}$$

   In this equation, R is the steady-state throughput, p is the loss rate, RTT is the average round-trip time, and MSS is the maximum segment size. You will need to devise strategies for estimating or determining these quantities using your traces.

   This section of the experiment will be documented and evaluated in your report. Be as detailed as possible in your report when describing the approach you used to estimate the values in the model equation. Discuss whether the model provides a good fit to the data and include graphs and/or tables to support your conclusions. If there is a discrepancy, suggest why the model might not provide a good fit. You should use information or insights from the articles by Mathis et al. [1] and Padhye et al. [2] to help with drawing these conclusions.

   BE CAREFUL WITH RTT CALCULATION – IT IS THE MOST COMMON SOURCE OF ERROR. IN ADDITION TO USING ANALYSIS OF THE TCP TRACE, CONSIDER USING A TOOL SUCH AS “ping” TO GET AN APPROXIMATE VALUE. THE VALUES YOU DERIVE FROM THESE TWO SOURCES SHOULD BE APPROXIMATELY THE SAME.

4 Important Dates and Evaluation

4.1 Demo

The demo will take place (remotely) in the lab session on Oct. 15 or 16 and will count for 2.5% of your final grade in the course. For the demo, you should be able to use Wireshark’s basic features (capture a trace, filter packets, etc.). You will also be asked to explain the TCP congestion window behavior of the traces you collected. Make sure you save and have available any necessary documentation and/or notes during the demo.

4.2 Report

The report is due at 23:59 on Oct. 19, and it will count for 2.5% of your final grade. Guidelines for preparing the report will be provided on myCourses. You should include a description of the experiments you conducted for the tasks in Section 3.2, including how you set up Wireshark, what files were downloaded from which servers, how long each trace is, how you computed the various statistics and quantities requested, and so on. Then, you should conduct the comparison with the Mathis et al. [1] model, and discuss any issues encountered in that section. Additional guidelines will be provided with the report template.

5 References

[1] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *ACM Computer Communication Review*, vol 27, no 3, July 1997.

[2] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," *Proceedings of ACM SIGCOMM*, Vancourver, Canada, September 1998.